BEWD - Models & Active Record

Matt Heath
Tech Lead, Platform
Hailo

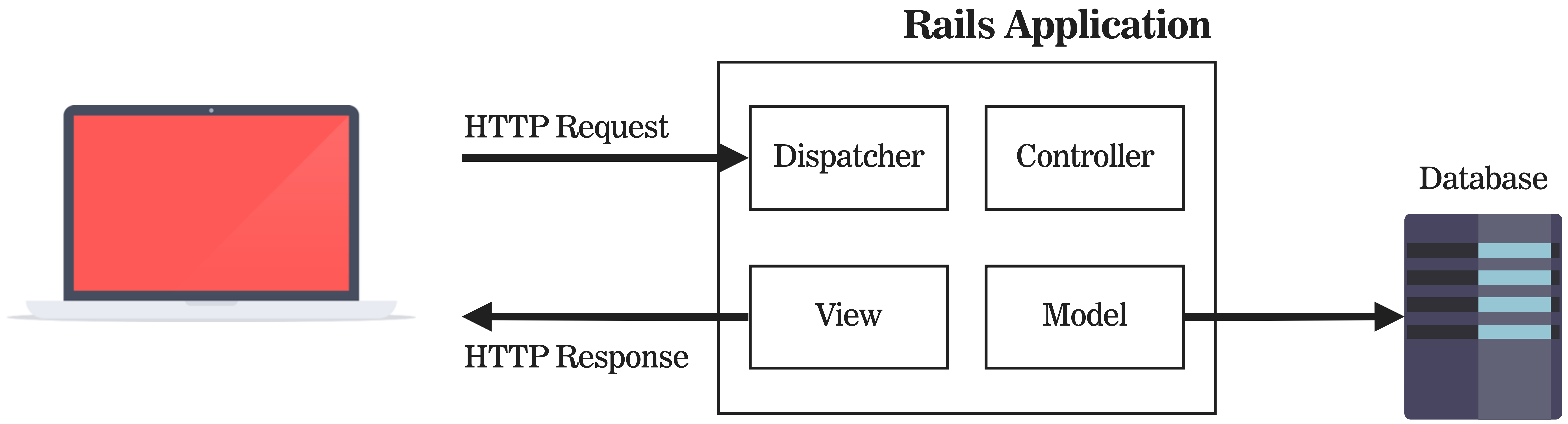
AGENDA

- Review
  - Quick Fire - Movies App
- Models
  - Databases
  - Generating Models
  - Migrations
  - seeds.rb
- Active Record
- Lab Time

Movies App - Quick Fire!

REVIEW

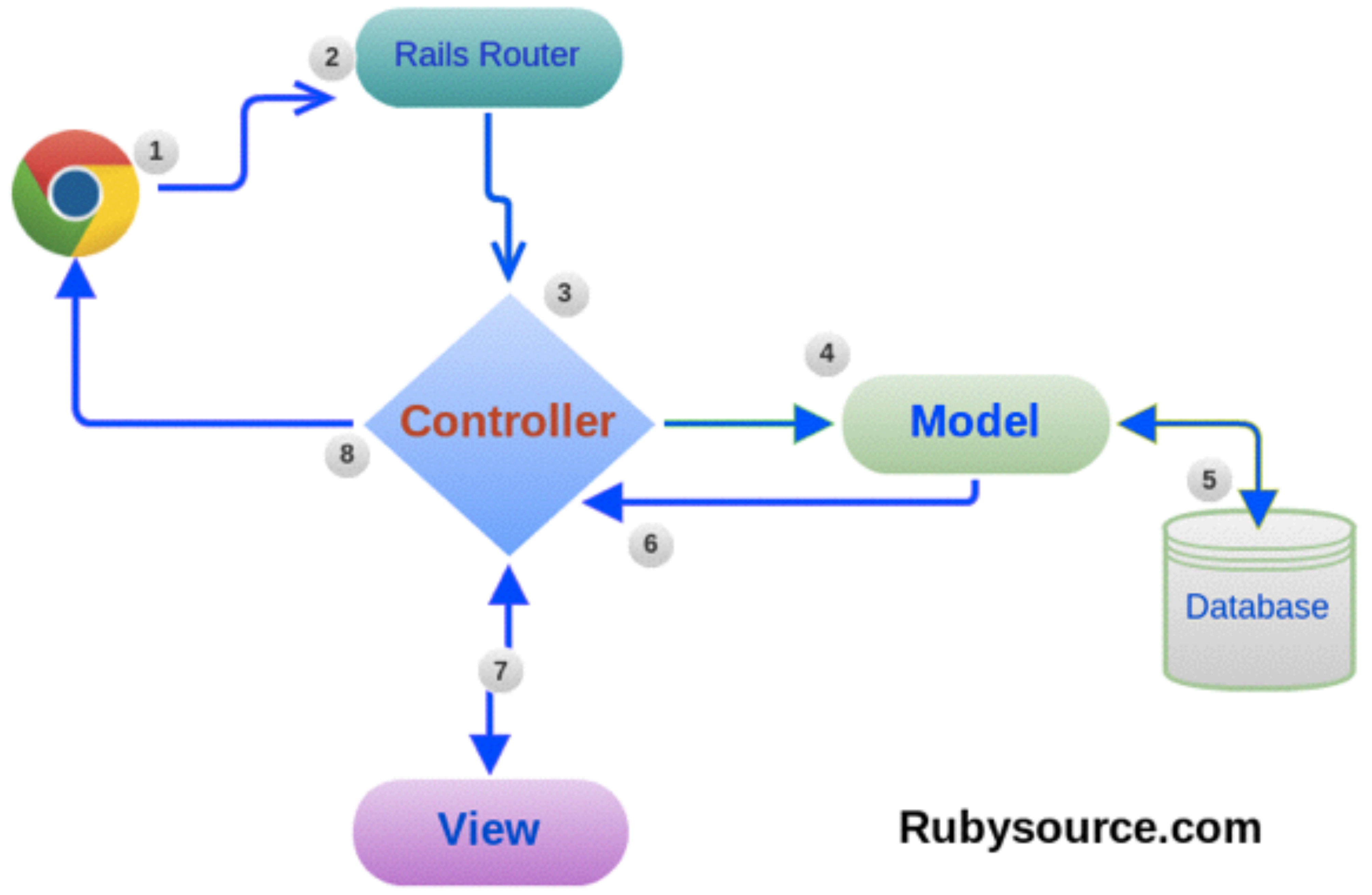
ROUTES, VIEWS & CONTROLLERS



REVIEW

MVC: MODEL, VIEW, CONTROLLER

- The Controller interacts with the Model
- The Controller renders the view, passing it Model data (using instance variables)
- The View and the Model do not interact



DATABASES

PERMANENT DATA STORAGE

- Permanent store for data (lives beyond a single request)
- Designed to handle data at scale (lots of data)
- Many different databases we can choose from, Rails handles almost all of them.

DATABASES

STANDARD DATA TYPES

» Text

» Numbers

» Dates / Times

» Booleans

DATABASES

TABLES

» Table: A database is made up of a collection of tables. The example below is a list of Employees.

id	first_name	age
1	John	29
2	Lina	24
3	Fred	46

DATABASES

SQL

- SQL: Structured Query Language
  A programming language used to search and save data to databases.

```
SELECT "movies".*
FROM   "movies"
WHERE  "movies"."title" = 'Jaws'
LIMIT  1
```

MODELS

TALKING TO THE DATABASE

» Models are needed to talk to the database

» We need to use the database to store *persistent* data (lives beyond a request lifecycle)

» Models simplify the task of working with a database

» Each model is used to talk to a specific table e.g. User model for Users table

» Rails models have special functionality to allow you to easily lookup data from the table, or make changes without having to use SQL directly

MODELS

CODE ALONG: SHIRTS

- Shirt Management app is an application we will build incrementally during class.
- The app allows users to manage their T-Shirts collection, by adding and deleting shirts to the database.
- For this lesson we will add a basic T-Shirt Model.

CODE ALONG

T-Shirt Management App
Models

RECAP

CREATE A NEW MODEL

```
rails g model Shirt name:string description:text
```

RECAP

RAKE

```
rake db:migrate
```

RECAP

RAKE

- Command line Task Runner for Ruby
- Used to:
  - Run Migrations
  - Seed your database

CODE ALONG

T-Shirt Management

RECAP

MIGRATION

```ruby
class AddImageToShirts < ActiveRecord::Migration
  def change
    add_column :shirts, :image, :string
  end
end
```

RECAP

MIGRATION

- Can add fields / columns to existing tables

```
# shortcut Syntax
rails g migration AddImageToShirts image:string
```

RECAP

WHAT CAN A MIGRATION DO?

» Add / remove columns from a table

» Modify columns in a table

» Add / remove tables

RECAP

SEEDS

» Fast and easy way to add data to your tables

» Place a `'seeds.rb'` file into your `'db/'` folder

» Run`'rake db:seed'`

CODE ALONG

T-Shirt Management

RECAP

ACTIVE RECORD

```
# Create
Shirt.create(name: "White Tee")

# Read
Shirt.find_by name: "White Tee"

# Update
my_shirt = Shirt.find_by name: "White Tee"
my_shirt.update description: "GA white Tee"

# Delete
my_shirt = Shirt.find_by name: "White Tee"
my_shirt.destroy
```

RECAP

ACTIVE RECORD

- Rails has a library called ActiveRecord to help Models talk to the database.

- Thus, Rails models are called ActiveRecord models.

- While ActiveRecord makes it easy to avoid SQL almost entirely, it's still valuable to know some SQL. Later in your development path, you will want to know which queries are more/less efficient so you can optimise them.

- For now though, we can enjoy the super-simple syntax of ActiveRecord to talk to our database.

MODELS

SUMMARY

» We want to persistently store our data,
so we need databases.

» Communicating with databases in SQL is complex, so we use ActiveRecord models to help us.

» ActiveRecord models are just Ruby Objects, so we can call methods on them and pass them around like any other object.

MODELS

SUMMARY

» Each Model *class* maps to a database table

```
>> User.all
>> User.create first_name: 'Matt'
>> User.find(1)
```

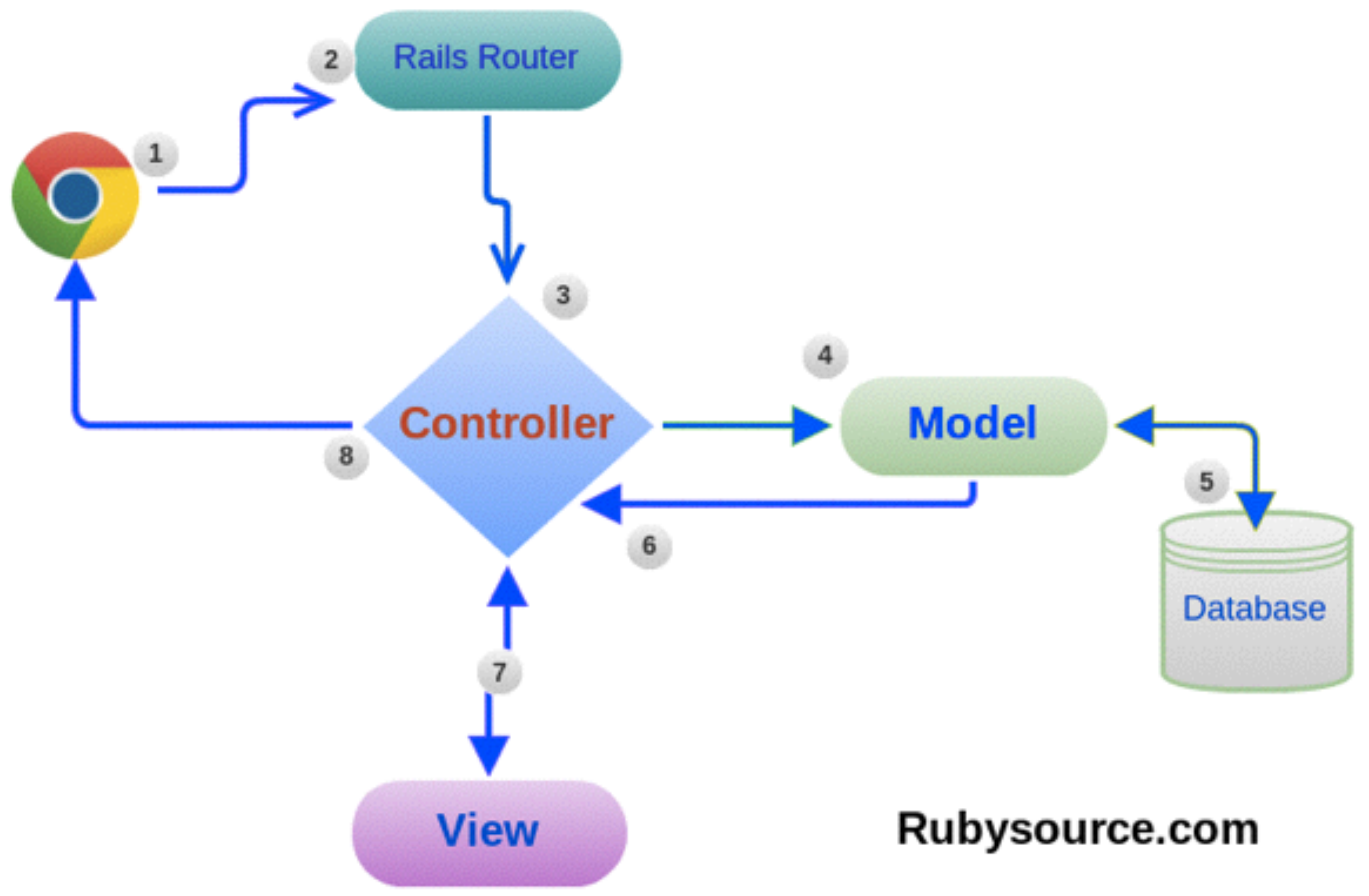
» Each Model *instance* maps to a single record in a table in the database

```
>> user = User.find(1)
=> #<User:0x007fcf8e9eebd8>
>> user.id
=> 1
>> user.update last_name: 'Heath'
```

MVC

MODEL, VIEW, CONTROLLER

- The Controller interacts with the Model
- The Controller renders the view, passing it Model data (using instance variables)
- The View and the Model do not interact



MVC

CONTROLLER

» The Controller interacts with the Model

```
# shirts_controller.rb

# Returns an array of Shirts (array of hashes)
@shirts = Shirts.all
```

MVC

CONTROLLER

» The Controller renders the view, passing it Model data

```
# shirts/index.html.erb

#Can be used in the view
@shirts
```

HOMEWORK

Complete and submit the Movies app (due next week)

Movies App - Movie Model